

Nomenclature

α, β	Learning rates of neural networks.
γ	Vector of all parameters in a deep kernel.
θ, \mathbf{p}	Coefficient, exponent parameter vectors of a GP.
$\theta^*, \mathbf{p}^*, \gamma^*$	Parameters that are trained on \mathcal{T}_* .
$\theta^e, \mathbf{p}^e, \gamma^e$	Parameters used to represent experience.
$\Delta\theta^*, \Delta\mathbf{p}^*$	Task-specific increments.
\hat{y}, \hat{s}^2	Estimations of mean and covariance from a GP model.
\mathcal{D}_i	Dataset collected from a related task $\mathcal{T}_i, i = 1, \dots, N$.
\mathcal{D}_m	Subsets sampled from \mathcal{D}_i for meta-learning.
\mathcal{D}_{mi}	The i^{th} subset $\mathcal{D}_m, m = 1, \dots, N_m$.
\mathcal{L}	Loss function.
$\mathcal{N}(\mu, \sigma^2)$	A normal distribution with mean μ and variance σ^2 .
\mathcal{S}, \mathcal{Q}	Support set and query set.
$\mathcal{S}_*, \mathcal{Q}_*$	Support set and query set collected from \mathcal{T}_* .
\mathcal{T}_*	Target optimization task.
\mathcal{T}_i	A related optimization task.
$\phi(\mathbf{w}, \mathbf{b})$	A neural network with weights \mathbf{w} and biases \mathbf{b} .
\mathbf{R}, \mathbf{r}	Correlation matrix, correlation vector of a GP model.
\mathbf{x}	Vector of decision variables, a sample/ solution.
\mathbf{x}_k^i	The k^{th} variable of the i^{th} sample \mathbf{x}^i .
\mathbf{y}	Vector of dependent variables.
$ \mathcal{D}_m $	Size of dataset \mathcal{D}_m .
$ \mathbf{R} $	The determinant of correlation matrix \mathbf{R} .
B	Number of related tasks in a batch (Batch size).
d	Number of decision variables/ Dimensionality of vector \mathbf{x} .
e	Mean square error value.
f	Objective function.
FE	Number of used expensive evaluations, a counter.
FE_{max}	Number of allowed expensive evaluations/ Budget.
h	Surrogate model.
$k(\mathbf{x}^i, \mathbf{x}^j \gamma)$	Kernel k with parameters γ and inputs $\mathbf{x}^i, \mathbf{x}^j$.
N	Number of related tasks.
n	Number of samples.
N_m	Number of subsets \mathcal{D}_m for meta-learning.
U	Number of update iterations in the meta-learning procedure.

A Discussion on Relevant Concepts

A.1 Difference between Experience-Based Optimization

In the past decade, experience-based optimization has attracted increasing attention as it uses the experience gained from other optimization problems to improve the optimization efficiency of target problems, mimicking human capabilities of cognition and knowledge generalization [15]. Optimization problems that provide experience or knowledge are regarded as source tasks, while those to be optimized are regarded as target tasks. To obtain useful experience, tasks related to target

tasks are chosen as source tasks, since they usually share domain-specific features with target tasks. Diverse experience-based optimization methods have been proposed to use the experience gained from related tasks to tackle target tasks. They can be divided into two categories according to the direction of experience transfer.

In the first category, experience is transferred mutually. Each considered optimization problem is a target task and also serves as one of the source tasks for other optimization problems. In other words, the roles of source and target tasks are compatible. One representative tributary is Evolutionary Multi-Tasking Optimization (EMTO) that aims to solve multiple optimization tasks concurrently [12, 47, 23, 3, 52]. In EMTO, experience is learned, updated, and spontaneously shared among target tasks through multi-task learning techniques. A variant of EMTO is multiforms optimization [15, 57, 14]. In multiforms optimization, multi-task learning methods are employed to learn experience from distinct formulations of a single target task.

In the second category, experience is transferred unidirectionally. The roles of source task and target task are not compatible, an optimization problem cannot serve as a source task and a target task simultaneously. One popular tributary is transfer optimization, which employs transfer learning techniques to transfer experience from source tasks to target tasks [37, 21, 20, 43]. In transfer learning, experience can be transferred from a single source task, multiple source tasks, or even source tasks from a different domain [60]. However, these transfer learning techniques pay more attention to experience transfer instead of experience learning. Despite the fact that diverse and complex situations of experience transfer have been studied [60], the difficulty of learning experience from small (expensive) source tasks has not been well studied. Actually, a common scenario in transfer learning is that the source task(s) is/are large enough such that useful experience can be obtained easily through solving the source task(s) [60]. In contrast to transfer optimization, recently, some experience-based optimization algorithms attempted to use meta-learning methods to learn experience from small source tasks, which are known as few-shot optimization (FSO)[50]. Since meta-learning only works for related tasks in the same domain, the situations of experience transfer are less complex than that of transfer learning. As a result, meta-learning pays more attention to experience learning instead of experience transfer. Domain-specific features are extracted as experience and no related task needs to be solved.

Our work falls into the FSO in the second category discussed above, since our experience is transferred unidirectionally. More importantly, our experience is learned across many related expensive tasks, rather than gained through solving a few or many source tasks. Therefore, our work is different from transfer optimization.

A.2 Difference between Bayesian Optimization and Surrogate-Assisted Evolutionary Algorithm

Bayesian Optimization (BO) and Surrogate-Assisted Evolutionary Algorithm (SAEA) are both model-based optimization methods for solving expensive optimization problems. The difference between BO and SAEA can be summarized as follows:

1. Type of surrogate models. BO uses probabilistic models, such as GPs, as surrogates. In comparison, SAEAs are flexible and can use any type of approximation model, not limited to probabilistic models.
2. Selection criterion. BO designs an acquisition function (AF) as the selection criterion for candidate solutions, explicitly considering the uncertainty in the probabilistic models. However, SAEAs do not necessarily account for model uncertainty. Instead, they focus on diversity and convergence as selection criteria, which can be implemented through separate functions.
3. Search algorithm. BO has no limitation on the search algorithm and can employ either gradient-based or gradient-free optimization techniques (such as EAs) to search candidate solutions. In contrast, SAEAs use only EAs as their underlying optimization algorithms.

As a result, there is some overlap between BO and SAEAs. A typical example is ParEGO, which employs GPs as its surrogates and designs an expected improvement (EI) function as its AF to consider uncertainty. Additionally, an EA is used as the underlying search algorithm.

Our FSEO framework focuses on meta-learning surrogates instead of AFs, making it compatible

with various SAEAs that do not rely on model uncertainty or AFs as selection criteria. In comparison, existing studies mainly work on the meta-learning of AFs, which limits their generality and applicability to SAEAs.

B Discussion on Framework Compatibility and Limitation

Our FSEO framework is applicable to regression-based SAEAs as our MDKL surrogates can be embedded in these SAEAs directly. Classification-based SAEAs are not compatible with our FSEO framework. The classification surrogates in these SAEAs are employed to learn the relation between pairs of solutions, or the relation between solutions and a set of reference solutions. The class labels used for surrogate training may fluctuate during optimization, making them difficult to learn across related tasks. Similarly, in ordinal-regression-based SAEAs, the ordinal relation values to be learned are not as stable as the fitness of expensive functions. So ordinal-regression-based SAEAs are also not compatible with our FSEO framework. In this paper, we focus on FSO for regression-based SAEAs, while other SAEA categories are left to be discussed in future work.

To the best of our knowledge, in existing studies, all meta-learning models used in the field of expensive optimization (not limited to multi-objective optimization) are regression-based models. Therefore, applying meta-learning models in classification-based or ordinal-regression-based optimization methods is a common limitation in the literature.

In addition, we admit that EMOPs are very complex and they have many subcategories. As a pioneer in few-shot optimization for EMOPs, it is impossible to consider all these subcategories in a single paper, so we follow the practice of existing studies [42] and consider the most common EMOPs in our work. For EMOPs with specific features, such as high-dimensional problems and dynamic problems, we plan to address them in future work, as EMOPs with special features are usually studied separately [32, 51].

C Generation of DTLZ variants

Our DTLZ optimization experiments generate m -objective DTLZ variants in the following ways:
DTLZ1:

$$f_1 = (a_1 + g)0.5 \prod_{i=1}^{m-1} x_i, \quad (8)$$

$$f_{j=2:m-1} = (a_j + g)(0.5 \prod_{i=1}^{m-j} x_i)(1 - x_{m-j+1}), \quad (9)$$

$$f_m = (a_m + g)0.5(1 - x_1), \quad (10)$$

$$g = 100 \left(k + \sum_{i=1}^k ((z_i - 0.5)^2 - \cos(20\pi(z_i - 0.5))) \right), \quad (11)$$

where \mathbf{z} is a vector consisting of the last $k = d - m + 1$ variables in \mathbf{x} . In other words, $\mathbf{z} = \{z_1, \dots, z_k\} = \{x_m, \dots, x_d\}$. The variants of DTLZ1 introduce only one variable $\mathbf{a} \in [0.1, 5.0]^m$ in Eq.(8), Eq.(9), and Eq.(10), where $\mathbf{a} = \mathbf{1}$ in the original DTLZ1. For out-of-range test, $\mathbf{a} \in [1.5, 5.0]^m$.

DTLZ2:

$$f_1 = (a_1 + g) \prod_{i=1}^{m-1} \cos\left(\frac{x_i \pi}{b_1}\right), \quad (12)$$

$$f_{j=2:m-1} = (a_j + g) \left(\prod_{i=1}^{m-j} \cos\left(\frac{x_i \pi}{b_j}\right) \right) \sin\left(\frac{x_{m-j+1} \pi}{b_j}\right), \quad (13)$$

$$f_m = (a_m + g) \sin\left(\frac{x_1 \pi}{b_m}\right), \quad (14)$$

$$g = \sum_{i=1}^k (z_i - 0.5)^2. \quad (15)$$

The variants of DTLZ2 introduce two variables $\mathbf{a} \in [0.1, 5.0]^m$ and $\mathbf{b} \in [0.5, 2.0]^m$ in Eq.(12), Eq.(13), and Eq.(14), where $\mathbf{a} = \mathbf{1}$ and $\mathbf{b} = \mathbf{2}$ in the original DTLZ2. For out-of-range test, $\mathbf{a} \in [1.5, 5.0]^m$, $\mathbf{b} \in [0.5, 1.5]^m$.

DTLZ3: The variants of DTLZ3 are generated in the same way as described in DTLZ2, except that the equation g in Eq.(15) is replaced by the one in Eq.(11).

DTLZ4: The variants of DTLZ4 are generated in the same way as described in DTLZ2, except that all x_i are replaced by $x_i^{1.00}$.

DTLZ5: The variants of DTLZ5 are generated in the same way as described in DTLZ2, except that all x_2, \dots, x_{m-1} are replaced by $\frac{1+2gx_i}{2(1+g)}$.

DTLZ6:

$$g = \sum_{i=1}^k z_i^{0.1}. \quad (16)$$

The variants of DTLZ6 are generated in the same way as described in DTLZ5, except that the equation g in Eq.(15) is replaced by the one in Eq.(16).

DTLZ7:

$$f_{j=1:m-1} = x_j + a_j, \quad (17)$$

$$f_m = (1 + g) \left(m - \sum_{i=1}^{m-1} \left(\frac{f_i}{1 + g} (1 + \sin(3\pi f_i)) \right) \right), \quad (18)$$

$$g = a_m + 9 \sum_{i=1}^k \frac{z_i}{k}. \quad (19)$$

The variants of DTLZ7 introduce one variable $\mathbf{a} \in [0.1, 5.0]^m$ in Eq.(17) and Eq.(19), where $a_{j=1:m-1} = 0$ and $a_m = 1$ in the original DTLZ7. For out-of-range test, $\mathbf{a} \in [1.5, 5.0]^m$.

D Performance of Meta-Learning Model and Component Contribution Analysis

Evaluating the effectiveness of learning experiences aims to demonstrate that our MDKL model can learn experience from related tasks and outperforms other meta-learning models. For this reason, the experiment is designed to answer the following questions:

- Given a small dataset \mathcal{S}_* from target task \mathcal{T}_* , can MDKL learn experience from related tasks and then generate a model that has the smallest MSE?
- If yes, which components of MDKL contribute to the effectiveness of learning experience? Meta-learning or/and deep kernel learning? If not, why not?

To answer the two questions above, we consider two experiments to evaluate the effectiveness of learning experience: amplitude prediction for unknown periodic sinusoid functions, and fuel consumption prediction for a gasoline motor engine. The former is a few-shot regression problem that motivates many meta-learning studies [13, 16, 41, 30], while the latter is a real-world regression problem [53].

D.1 Meta-Learning Model Experiment on Sinusoid Function Regression Problem

In the sinusoid regression experiment, we learn experience from a series of 1-dimensional sinusoid functions:

$$y = A \sin(wx + b) + \epsilon, \quad (20)$$

where the amplitude A and phase w of sine waves are varied between functions. The target is to approximate an unknown sinusoid function with a small support dataset \mathcal{S}_* and the learned experience. Clearly, by integrating experience with \mathcal{S}_* , we estimate parameters (A, w, b) for an unknown sinusoid function. As a result, the output y of the given sinusoid function can be predicted once a query data x is inputted.

D.1.1 Generation of Sinusoid Function Variants

As suggested in [13, 16], we set amplitude $A \in [0.1, 5.0]$, frequency $w \in [0.999, 1.0]$, phase $b \in [0, \pi]$, and Gaussian noise $\epsilon \sim (0, 0.1)$. Therefore, a sinusoid function can be generated by sampling three parameters (A, w, b) from their ranges uniformly. In total, $N_m = N = 20000$ related sinusoid functions are generated at random.

D.1.2 Experimental Setups

All data points x are sampled from the range $\in [-5.0, 5.0]$. In the meta-learning procedure, both support set and query set contain 5 data points. Hence, a dataset \mathcal{D}_i is sampled from each (related) sinusoid function \mathcal{T}_i , and $|\mathcal{D}_i| = |\mathcal{D}_m| = 10$. Six experiments are conducted, with $|\mathcal{S}_*| = \{2, 3, 5, 10, 20, 30\}$ data points are sampled from the target function in each experiment. Considering Gaussian noise ϵ could be relatively large when amplitude A is close to 0.1, normalized mean squared error (NMSE) is chosen as a performance indicator. NMSE is measured using a dataset that contains 100 data points sampled uniformly from the x range.

D.1.3 Comparison Methods

In this experiment, three families of modeling methods are compared with our MDKL model:

- **Meta-learning methods** that were proposed for regression tasks: MAML [13], ALPaCA [16], and DKT [30]. The configurations of MAML, ALPaCA, and DKT are the same as suggested in the original literature.
- **Non-meta-learning method** that is widely used for regression tasks: the GP model. We choose a GP as a baseline since it is effective and more relevant to MDKL than other non-meta-learning modeling methods. We set the range of base kernel parameters in the GP model as $\theta \in [10^{-5}, 10]$ and $p \in [1, 2]$.
- **MDKL related methods** that are designed to investigate which components of MDKL contribute to its modeling performance: GP_Adam, DKL, and MDKL_NN. GP_Adam is a GP model fitted by the Adam optimizer. The combination of GP_Adam and a neural network results in a kind of DKL algorithm. MDKL_NN is a meta-learning version of DKL, but it learns only neural network parameters through meta-learning and does not have task-independent base kernel parameters.

D.1.4 Meta-Learning Results and Component Contribution Analysis

Table 3 reports the statistical test results of the NMSE values achieved by comparison algorithms in sinusoid function regression experiments. Each row shows the results obtained using the same number of fitness evaluations $|\mathcal{S}_*|$ for model training. The results of the Wilcoxon rank sum test between MDKL and the other compared algorithms are listed in the last row. It can be observed that both MDKL and DKT achieved the lowest NMSE values on all tests compared to other meta-learning and non-meta-learning modeling methods.

Contributions of MDKL components are analyzed through statistical tests between MDKL related methods. The statistical test results between DKL and GP_Adam are 5/1/0, showing that DKL is preferable to GP_Adam when only a few data points are available for modeling. Hence, using a neural network to build a deep kernel for GP can enhance the performance of modeling. When the meta-learning technique is applied to DKL, the statistical test results between MDKL_NN and

Table 3: Mean NMSE and standard deviation (in parentheses) of 30 runs on the amplitude regression of sinusoid function. GP [36] is a widely used surrogate in SAEAs, MAML [13], ALPaCA [16], and DKT [30] are meta-learning methods. GP_Adam is a GP model fitted by the Adam optimizer. DKL is a deep kernel learning algorithm that adds a neural network to GP_Adam. MDKL_NN applies meta-learning to DKL, but no task-independent base kernel parameters are shared between related tasks. Support data points are used to train non-meta surrogates or adapt meta-learning surrogates. ‘+’, ‘ \approx ’, and ‘-’ denote MDKL is statistically significantly superior to, almost equivalent to, and inferior to the compared modelling methods in the Wilcoxon rank sum test (significance level is 0.05), respectively. The last row counts the total win/tie/loss results. It shows that MDKL and DKT have lower NMSE than other models. The effectiveness of meta-learning on both the neural network and the base kernel has been demonstrated on this example.

Support data points $ S_* $	GP	GP_Adam	DKL	MDKL_NN	MDKL (ours)	DKT	MAML	ALPaCA
2	1.63e-1(9.18e-2) \approx	1.93e-1(9.72e-2)+	1.63e-1(9.05e-2) \approx	1.57e-1(9.26e-2) \approx	1.56e-1(9.49e-2)	1.56e-1(9.49e-2) \approx	2.09e-1(3.63e-1) \approx	1.07e+0(2.57e+0) \approx
3	1.27e-1(6.04e-2) \approx	1.62e-1(6.53e-2)+	1.21e-1(5.96e-2) \approx	1.16e-1(5.95e-2) \approx	1.10e-1(6.20e-2)	1.10e-1(6.20e-2) \approx	2.09e-1(3.60e-1) \approx	4.36e-1(8.57e-1) \approx
5	6.76e-2(4.62e-2) \approx	1.09e-1(5.61e-2)+	7.52e-2(4.40e-2)+	6.38e-2(3.91e-2) \approx	4.79e-2(3.73e-2)	4.79e-2(3.70e-2) \approx	2.08e-1(3.59e-1)+	4.31e-1(8.04e-1) \approx
10	1.70e-2(1.87e-2) \approx	6.13e-2(4.58e-2)+	2.87e-2(1.89e-2)+	1.89e-2(1.61e-2)+	1.07e-2(1.16e-2)	1.09e-2(1.17e-2) \approx	2.08e-1(3.58e-1)+	6.59e-1(2.14e+0)+
20	5.42e-3(7.64e-3)+	3.92e-2(4.29e-2)+	9.64e-3(1.02e-2)+	5.24e-3(6.57e-3)+	2.57e-3(4.53e-3)	2.63e-3(4.61e-3) \approx	2.08e-1(3.58e-1)+	1.13e-1(3.39e-1)+
30	3.97e-3(7.40e-3)+	3.32e-2(4.18e-2)+	4.81e-3(6.68e-3)+	3.20e-3(5.85e-3)+	1.68e-3(3.61e-3)	1.60e-3(3.39e-3) \approx	2.08e-1(3.58e-1)+	7.59e-2(2.01e-1)+
+ / \approx / -	2/4/0	6/0/0	4/2/0	3/3/0	-/-	0/6/0	4/2/0	3/3/0

DKL are 3/3/0. Meta-learning of neural network parameters is necessary, since it contributes to the performance of MDKL. Further statistical tests between MDKL and MDKL_NN yield results of 3/3/0, indicating that the meta-learning of the base kernel parameters is effective for this regression problem.

D.2 Meta-Learning Model Experiment on Real-World Engine Performance Regression Problem

In this subsection, we focus on a Brake Special Fuel Consumption (BSFC) regression task for a gasoline motor engine [53], where BSFC is evaluated on a gasoline engine simulation (denoted as \mathcal{T}_*).

D.2.1 Experimental Setups

The related tasks \mathcal{T}_i used in our experiment are $N = 100$ gasoline engine models. These engine models have different behaviors when compared with \mathcal{T}_* , but they share the basic features of gasoline engines. All related tasks and the target task have the same six decision variables. Each related task \mathcal{T}_i provides only 60 solutions, forming a dataset \mathcal{D}_i . The size of datasets used for meta-learning $|\mathcal{D}_m|$ is set to 40. Six tests are performed with $|S_*| = \{2, 3, 5, 10, 20, 40\}$ data points are sampled from the real engine simulation \mathcal{T}_* in each test. MSE is chosen as an indicator of modeling accuracy, which is measured using a dataset consisting of 12500 data points that are sampled uniformly from the engine decision space. The comparison algorithms are the same as described in Appendix D.1.

D.2.2 Meta-Learning Results and Component Contribution Analysis

The statistical test results of the MSE values achieved by the comparison algorithms in BSFC regression experiments are summarized in Table 4. Each row shows the results obtained using the same number of fitness evaluations $|S_*|$ for model training. The results of the Wilcoxon rank sum test between MDKL and the other compared algorithms are listed in the last row. It can be observed that MDKL and MDKL_NN outperform the other comparison modeling methods, since they have achieved the lowest MSE on all tests.

Table 4: Mean MSE and standard deviation (in parentheses) of 30 runs on the regression of engine fuel consumption. Support data points are used to train non-meta surrogates or adapt meta-learning surrogates. All results are normalized since the actual engine data is unable to be disclosed.

Support data points $ S_* $	GP	GP_Adam	DKL	MDKL_NN	MDKL (ours)	DKT	MAML	ALPaCA
2	2.23e+1(3.20e+0)+	2.37e+1(6.30e+0)+	2.30e+1(5.87e+0)+	1.75e+1(6.33e+0) \approx	1.72e+1(6.34e+0)	1.81e+1(5.68e+0) \approx	1.87e+1(6.37e+0) \approx	1.91e+1(1.02e+1) \approx
3	2.14e+1(3.74e+0)+	2.41e+1(1.38e+1)+	2.20e+1(3.74e+0)+	1.45e+1(7.13e+0) \approx	1.45e+0(7.01e+0)	1.55e+1(6.66e+0) \approx	1.80e+1(4.69e+0) \approx	2.13e+1(1.97e+1) \approx
5	2.13e+1(3.27e+0)+	2.46e+1(1.00e+1)+	2.07e+1(3.95e+0)+	1.12e+1(6.65e+0) \approx	1.10e+1(6.58e+0)	1.21e+1(6.49e+0) \approx	1.84e+1(6.05e+0)+	1.99e+1(2.29e+1)+
10	1.84e+1(1.89e+0)+	2.06e+1(1.19e+1)+	2.10e+1(5.79e+0)+	7.19e+0(4.82e+0) \approx	7.08e+0(4.77e+0)	7.99e+0(4.87e+0) \approx	1.70e+1(5.54e+0)+	1.38e+1(8.12e+0)+
20	1.56e+1(2.00e+0)+	2.38e+1(1.05e+1)+	1.76e+1(2.42e+0)+	5.03e+0(1.82e+0) \approx	4.86e+0(1.71e+0)	5.74e+0(1.91e+0)+	1.50e+1(2.59e+0)+	1.01e+1(5.52e+0)+
40	1.28e+1(2.03e+0)+	1.48e+1(7.35e+0)+	1.67e+1(3.73e+0)+	4.13e+0(7.90e-1) \approx	4.00e+0(8.59e-1)	4.92e+0(1.09e+0)+	1.45e+1(1.85e+0)+	8.01e+0(3.35e+0)+
+ / \approx / -	6/0/0	6/0/0	6/0/0	0/6/0	-/-	2/4/0	4/2/0	4/2/0

Additional Wilcoxon rank sum tests have been conducted between MDKL related algorithms to answer our second question (results are not reported in Table 4). The statistical test results between DKL and GP_Adam are 1/5/0, indicating that the neural network in DKL makes some contributions to the performance of MDKL. The statistical test results between MDKL_NN and DKL are 6/0/0, demonstrating that the meta-learning of neural network parameters constructs a useful deep kernel and contributes to the improvement of modeling accuracy. However, there is no significant difference between the performance of MDKL and that of MDKL_NN, the meta-learning on base kernel parameters does not play a critical role in this engine problem. In comparison, the meta-learning on base kernel parameters is effective in sinusoid function regression experiments (see Appendix D.1). In addition, the statistical test results between MDKL_NN and MAML are 4/2/0. Considering that MAML is a neural network regressor learned through meta-learning, we can conclude that GP is an essential component of our MDKL. In summary, all components of MDKL are necessary and contribute to the effectiveness of learning experience.

The comparison experiments on sinusoid functions and the gasoline motor engine have demonstrated the effectiveness of our MDKL modeling method in the learning of experience. Given a small dataset of the target task, the model learned through MDKL method has the lowest MSE among all comparison models. Additionally, the investigation between MDKL and its variants shows that all components in MDKL have contributed to the effectiveness of learning experience. However, similar to other meta-learning studies [13, 16], we have not defined the similarity between tasks. In other words, the boundary between related and unrelated tasks has not been defined. This should be a topic of further study on meta-learning. Moreover, the relationship between task similarity and modeling performance has not been investigated. Instead, we study the relationship between task similarity and SAEA optimization performance in Section 5.2.1, since our main focus is surrogate-assisted evolutionary optimization.

E Additional Details on Expensive Multi-Objective Optimization

E.1 Experimental Setups

For all meta-learning methods used in our experiments, their basic setups are listed in Table 5. The neural network structure is suggested by [13, 30], and the learning rates are the default values that have been widely used in many meta-learning methods [16, 30].

Table 5: Parameter setups for meta-learning methods.

Module	Parameter	Value
Meta-learning	Number of meta-learning datasets N_m	20000
	Number of update iterations U	2000
	Batch size B	10
Neural network	Number of hidden layers	2
	Number of units in each hidden layer	40
	Learning rates α, β	0.001, 0.001
	Activation function	ReLU

The parameter setups for this multi-objective optimization experiment are listed in Table 6.

Table 6: Parameter setups for DTLZ optimization.

Parameter	MOEA/D-FS	Comparisons
Number of related tasks N	20000 (N_m in Table 5)	-
Size of datasets from related tasks $ D_i $	20 (2d)	-
Size of datasets for meta-learning $ D_m $	$ D_i $	-
Evaluations for initialization	10 (1d)	100 (10d)
Evaluations for further optimization	50	50
Total evaluations	60	150

E.2 Comparison Algorithms

As explained in Appendix B, our FSEO framework is compatible with regression-based SAEAs. Hence, we select MOEA/D-EGO [58] as an example and replace its GP surrogates with our MDKL surrogates. The resulting algorithm is denoted as MOEA/D-FS. Note that it is not necessary to specially select a newly proposed regression-based SAEA as our example, our work mainly aims to save evaluations with experience and observe if there is any damage to the optimization performance

caused by the saving of evaluations. Therefore, it does not make any difference which regression-based SAEA or BO we choose as our example.

To demonstrate that the improvement of optimization performance caused by using experience on DTLZ problems is significant, several state-of-the-art SAEAs and MOBO are also compared as baselines, including qLogEHVI [1], DirHVEI [59], KMOEA-TIC [31], ESBCEO [4], KTA2 [35], qEHVI [10], OREA [54], and K-RVEA [9]. Among these algorithms, K-RVEA, KTA2, and KMOEA-TIC use regression-based surrogates, OREA employs an ordinal-regression-based surrogate, qLogEHVI, DirHVEI, ESBCEO, and qEHVI are MOBO.

We implemented the FSEO framework and MOEA/D-EGO, while the code of K-RVEA, KTA2, ESBCEO, and DirHVEI is available on PlatEMO [39], an open source MATLAB platform for evolutionary multi-objective optimization. The code of qLogEHVI and qEHVI is implemented in BoTorch, a public Python library. The code of OREA and KMOEA-TIC is obtained from their authors. To make a fair comparison, all comparison algorithms share the same initial dataset \mathcal{S}_* in an independent run. We also set $\theta \in [10^{-5}, 100]^d$ and $\mathbf{p} = \mathbf{2}$ for all GP surrogates as suggested in [35], these GP surrogates are implemented through DACE [33]. Other configurations are the same as suggested in the original literature.

E.3 Result Table and Analysis of Expensive Multi-Objective Optimization

The experience learned from related tasks makes MOEA/D-EGO more competitive when compared to other SAEAs and MOBO algorithms. By using MDKL surrogates, significant improvements of optimization results in terms of the IGD+ value are achieved on DTLZ1, DTLZ2, DTLZ3, and DTLZ5. As a result, MOEA/D-FS achieves the smallest IGD+ values on DTLZ2 and DTLZ3, and its optimization results on DTLZ1 and DTLZ5 are much closer to the best optimization results (e.g. results obtained by qEHVI and OREA) than MOEA/D-EGO. Although MOEA/D-FS does not achieve the smallest IGD+ values on all DTLZ problems, it should be noted that MOEA/D-FS is still the most competitive algorithm among comparison algorithms due to its overall performance and few cost of expensive evaluations. Table 7 shows that only qLogEHVI and qEHVI outperform MOEA/D-FS on at least three DTLZ problems, while MOEA/D-FS outperforms all other comparison algorithms on at least three DTLZ problems. (An additional comparison between MOEA/D-FS and qLogEHVI, qEHVI, is reported in Table 10, where the optimization results are compared when the same expensive evaluation budgets are available for all comparison algorithms.) Furthermore, the IGD+ values of MOEA/D-FS are achieved with an evaluation budget of 60, while the IGD+ values of other SAEAs and MOBO algorithms are reached with a cost of 150 evaluations (see Table 6).

Table 7: Mean IGD+ values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS and the comparison algorithms initialize their surrogates with 10, 100 samples, respectively. Extra 50 evaluations are allowed in the further optimization. ‘+’, ‘ \approx ’, and ‘-’ denote MOEA/D-FS is statistically significantly superior to, equivalent to, and inferior to the compared algorithms in the Wilcoxon rank sum test (significance level is 0.05), respectively. The last row is the total win/tie/loss results. Competitive results can be observed from the comparisons between MOEA/D-FS and MOEA/D-EGO, while 90 evaluations are saved using the experience learned from related tasks.

Problems	MOEA/D-EGO	MOEA/D-FS	qLogEHVI	DirHVEI	KMOEA-TIC	ESBCEO	qEHVI	KTA2	OREA	K-RVEA
DTLZ1	1.07e+2(2.05e+1)+	9.70e+1(1.87e+1)	6.94e+1(1.07e+1)-	1.04e+2(2.47e+1) \approx	1.10e+2(2.29e+1)+	8.81e+1(1.18e+1) \approx	7.60e+1(7.88e+0)-	1.01e+2(2.34e+1) \approx	1.02e+2(1.97e+1) \approx	1.18e+2(2.41e+1)+
DTLZ2	2.99e-1(7.01e-2)+	1.43e-1(2.29e-2)	1.51e-1(1.49e-2)+	2.13e-1(4.58e-2)+	2.10e-1(7.10e-2)+	3.39e-1(3.78e-2)+	1.55e-1(1.65e-2)+	2.14e-1(3.78e-2)+	1.76e-1(4.69e-2)+	2.69e-1(5.87e-2)+
DTLZ3	3.15e+2(6.04e+1)+	1.97e+2(1.64e+1)	1.92e+2(1.23e+1) \approx	2.86e+2(5.98e+1)+	2.98e+2(6.14e+1)+	2.09e+2(4.23e+1) \approx	1.89e+2(2.78e+1) \approx	2.67e+2(6.58e+1)+	2.72e+2(6.88e+1)+	3.24e+2(5.80e+1)+
DTLZ4	5.04e-1(8.25e-2) \approx	4.44e-1(1.34e-1)	2.94e-1(9.29e-2)-	4.77e-1(1.11e-1) \approx	4.26e-1(9.19e-2) \approx	4.99e-1(7.37e-2) \approx	2.93e-1(1.01e-1)-	4.51e-1(9.38e-2) \approx	3.18e-1(1.54e-1)-	4.57e-1(1.12e-1) \approx
DTLZ5	2.39e-1(7.17e-2)+	1.13e-1(2.24e-2)	1.27e-1(2.62e-2)+	1.36e-1(3.72e-2)+	8.73e-2(2.77e-2)-	2.68e-1(3.62e-2)+	1.22e-1(2.29e-2)+	1.44e-1(4.52e-2)+	7.84e-2(2.42e-2)+	1.92e-1(5.87e-2)+
DTLZ6	1.29e+0(4.74e-1) \approx	1.11e+0(5.70e-1)	3.28e-1(2.37e-1)-	2.65e-1(2.16e-1)-	2.90e+0(5.34e-1)+	2.41e+0(7.97e-1)+	3.47e-1(2.50e-1)-	3.37e+0(6.49e-1)+	4.60e+0(1.19e+0)+	4.62e+0(6.31e-1)+
DTLZ7	3.31e-1(3.11e-1)-	2.47e+0(1.89e+0)	2.51e-1(1.04e-1)-	1.18e-1(2.85e-2)-	9.44e-2(1.23e-2)-	5.47e-1(2.46e-1)-	5.65e-1(2.94e-1)-	4.34e-1(2.16e-1)-	2.14e+0(1.15e+0) \approx	1.74e-1(3.51e-2)-
+/-/ \approx -	4/2/1	-/-/-	2/1/4	3/2/2	4/1/2	3/3/1	2/1/4	4/2/1	3/2/2	5/1/1

E.4 Experimental Results in IGD and HV Metrics

The performance of our method and the comparison algorithms are also evaluated on inverted generational distance (IGD) [5] and Hypervolume (HV) [61] metrics.

Results in terms of IGD values are reported in Table 8 and Fig. 5. A smaller IGD value indicates a better optimization result.

Results in terms of HV values are reported in Table 9 and Fig. 6. A larger HV value indicates a better optimization result.

Table 8: Mean IGD values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS and comparison algorithms initialize their surrogates with 10, 100 samples, respectively. Extra 50 evaluations are allowed in the further optimization.

Problems	MOEA-D-EGO	MOEA-D-FS	qLogEHVI	DirHVEI	KMOEATIC	ESBCEO	qEHVI	KTA2	OREA	K-RVEA
DTLZ1	1.07e+2(2.05e+1)+	9.70e+1(1.87e+1)	6.94e+1(1.07e+1)	1.04e+2(2.47e+1)	1.10e+2(2.29e+1)+	8.81e+1(1.18e+1)	7.60e+1(7.88e+0)	1.01e+2(2.34e+1)	1.02e+2(1.97e+1)	1.18e+2(2.41e+1)+
DTLZ2	3.30e-1(7.23e-2)+	1.72e-1(2.41e-2)	2.24e-1(2.73e-2)+	3.40e-1(3.62e-2)+	2.86e-1(6.31e-2)+	3.64e-1(3.29e-2)+	2.35e-1(2.84e-2)+	2.45e-1(3.57e-2)+	2.14e-1(4.10e-2)+	3.08e-1(4.93e-2)+
DTLZ3	3.15e+2(6.04e+1)+	1.97e+2(1.64e+1)	1.92e+2(3.23e+1)	2.86e+2(5.98e+1)+	2.98e+2(6.14e+1)+	2.09e+2(4.23e+1)	1.89e+2(2.78e+1)	2.67e+2(6.58e+1)+	2.72e+2(6.88e+1)+	3.24e+2(5.80e+1)+
DTLZ4	7.51e-1(1.50e-1)	7.96e-1(2.25e-1)	3.94e-1(0.69e-2)	6.87e-1(9.73e-2)	5.23e-1(8.60e-2)	6.70e-1(8.05e-2)	3.92e-1(9.71e-2)	6.30e-1(1.51e-1)	5.64e-1(2.01e-1)	5.94e-1(1.23e-1)
DTLZ5	2.47e-1(7.21e-2)+	1.17e-1(2.08e-2)	2.31e-1(4.51e-2)+	2.00e-1(5.26e-2)+	1.18e-1(3.17e-2)	2.83e-1(3.00e-2)+	2.19e-1(3.70e-2)+	1.61e-1(4.60e-2)	8.64e-2(2.48e-2)	2.13e-1(5.55e-2)+
DTLZ6	1.36e+0(4.10e-1)	1.18e+0(5.35e-1)	6.06e-1(2.54e-1)	5.29e-1(2.25e-1)	2.92e+0(5.35e-1)	2.45e+0(7.92e-1)+	6.18e-1(2.32e-1)	3.37e+0(6.50e-1)+	4.61e+0(1.18e+0)	4.63e+0(6.26e-1)+
DTLZ7	4.22e-1(3.16e-1)	2.56e+0(1.86e+0)	3.72e-1(8.17e-2)	2.22e-1(3.76e-2)	1.85e-1(1.81e-2)	6.21e-1(2.43e-1)	6.88e-1(2.67e-1)	5.54e-1(2.38e-1)	2.21e+0(1.11e+0)	2.55e-1(4.36e-2)
+/-	2/1/1	2/1/1	2/1/1	3/1/3	4/1/2	3/2/2	2/1/4	4/1/2	3/2/2	5/0/2

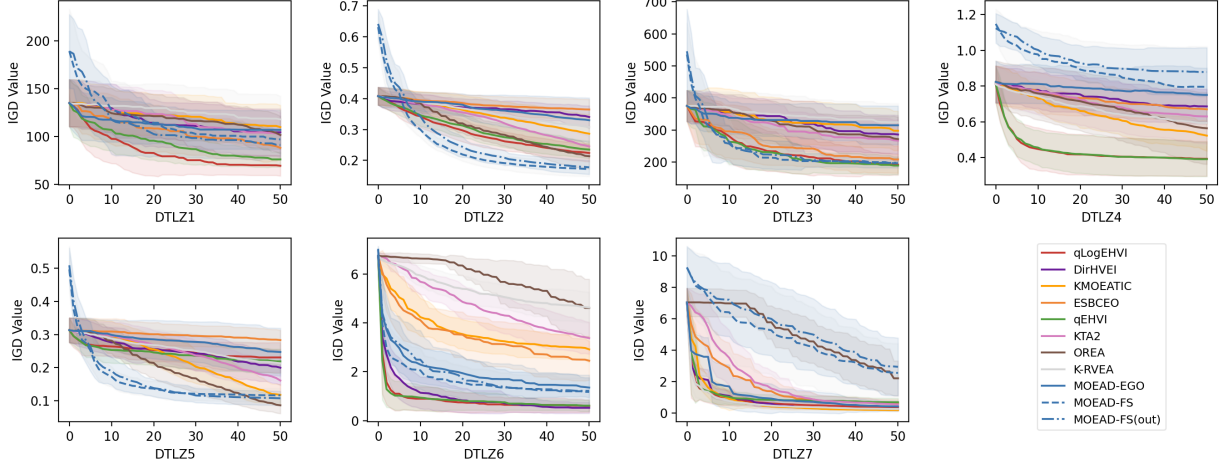


Figure 5: IGD curves averaged over 30 runs on 7 DTLZ problems. Solid lines are mean values, while shadows are error regions. MOEA/D-FSs and comparison algorithms initialize their surrogates with 10, 100 samples, respectively. Extra 50 evaluations are allowed in the further optimization. Note that ‘FS(out)’ indicates the target task is excluded from the range of related tasks during the meta-learning procedure). X-axis denotes the number of evaluations used after the surrogate initialization.

Table 9: Mean HV values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS and comparison algorithms initialize their surrogates with 10, 100 samples, respectively. Extra 50 evaluations are allowed in the further optimization.

Problems	MOEA-D-EGO	MOEA-D-FS	qLogEHVI	DirHVEI	KMOEATIC	ESBCEO	qEHVI	KTA2	OREA	K-RVEA
DTLZ1	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)
DTLZ2	2.02e-1(1.28e-1)+	4.69e-1(3.70e-2)	5.16e-1(3.03e-2)	2.25e-1(1.04e-1)+	2.91e-1(1.29e-1)+	1.39e-1(4.55e-2)+	5.10e-1(3.81e-2)	3.19e-1(6.49e-2)+	3.80e-1(7.64e-2)+	1.93e-1(8.93e-2)+
DTLZ3	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)
DTLZ4	6.25e-2(5.53e-2)+	1.43e-1(7.17e-2)	1.70e-1(3.30e-1)	5.66e-2(5.56e-2)+	6.50e-2(7.07e-2)+	2.27e-2(2.65e-2)+	1.69e-1(1.16e-1)	6.49e-2(7.42e-2)+	2.11e-1(1.37e-1)	3.81e-2(4.24e-2)+
DTLZ5	4.50e-2(4.17e-2)+	1.63e-1(1.60e-2)	8.16e-2(3.67e-2)+	1.08e-1(4.65e-2)+	1.58e-1(3.69e-2)	1.64e-2(1.42e-2)+	8.12e-2(3.32e-2)+	7.98e-2(3.80e-2)+	1.49e-1(2.88e-2)	4.82e-2(2.78e-2)+
DTLZ6	1.24e-3(3.77e-3)	1.59e-2(3.46e-2)	1.07e-1(3.17e-2)	9.00e-2(5.84e-2)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	8.94e-2(5.86e-2)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)
DTLZ7	3.83e-1(8.50e-2)	1.12e-1(1.27e-1)	3.07e-1(7.10e-2)	5.11e-1(8.24e-2)	4.67e-1(1.27e-2)	1.71e-1(8.33e-2)	1.92e-1(1.01e-1)	3.70e-1(3.88e-2)	2.97e-2(4.45e-2)	3.79e-1(2.61e-2)
+/-	3/3/1	2/1/1	1/3/3	3/2/2	2/4/1	3/3/1	1/3/3	3/3/1	1/6/0	3/3/1

E.5 Performance on Expensive Multi-Objective Optimization Under the Same Evaluation Budget

The statistical test results reported in the last row of Table 7 show that qLogEHVI [1], DirHVEI [59], qEHVI [10], and OREA [54] are the best four comparison algorithms when compared with our MOEA/D-FS. In this subsection, we evaluate the performance of MOEA/D-FS when no extra evaluation is saved. For this purpose, we compare the optimization performance of these five comparison algorithms under the same evaluation budget: 10 evaluations (1d) for surrogate initialization and 50 evaluations for further optimization. The statistical test results are reported in Table 10. It can be seen that our MOEA/D-FS is competitive to the state-of-the-art MOBO and SAEAs when only 1d evaluations are used to initialize their surrogates. Considering that the underlying optimization algorithm in our example is MOEA/D-EGO, a classic algorithm that was proposed a decade ago, the effectiveness of our FSEO framework in assisting existing optimization algorithms has been demonstrated. Note that OREA is an evolutionary algorithm assisted by ordinal-regression-based surrogates. Currently, our FSEO framework is applicable to the SAEAs working with regression-based surrogates. The meta-learning of ordinal-regression models can be a topic of further research.

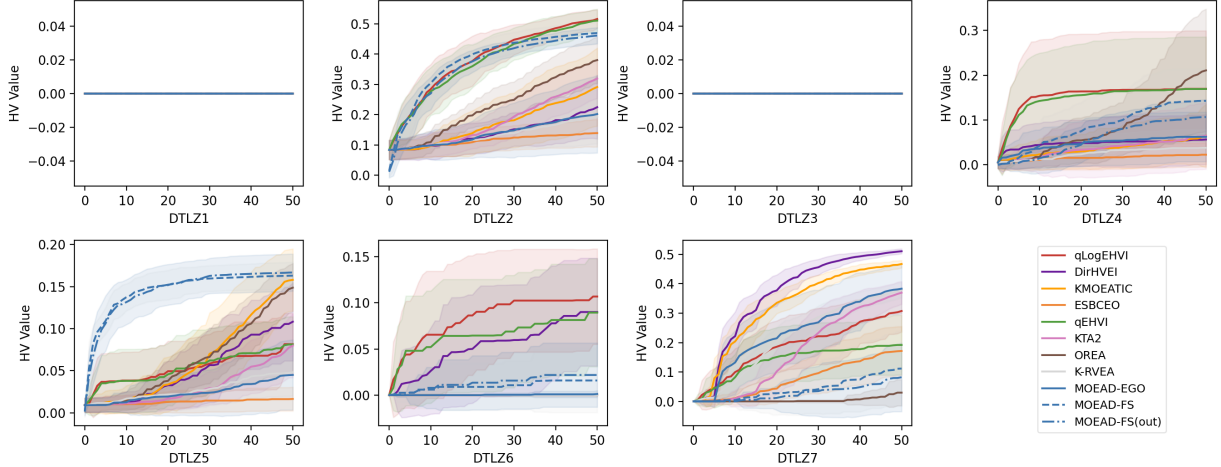


Figure 6: HV curves averaged over 30 runs on 7 DTLZ problems. Solid lines are mean values, while shadows are error regions. MOEA/D-FSs and comparison algorithms initialize their surrogates with 10, 100 samples, respectively. Extra 50 evaluations are allowed in the further optimization. Note that ‘FS(out)’ indicates the target task is excluded from the range of related tasks during the meta-learning procedure). X-axis denotes the number of evaluations used after the surrogate initialization.

Table 10: Mean IGD+ values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS is compared with qLogEHVI, DirHVEI, qEHVI, and OREA under the same evaluation budget: 10 evaluations for surrogate initialization and 50 evaluations for the optimization process.

Problems	MOEA-D-FS	qLogEHVI	DirHVEI	qEHVI	OREA
DTLZ1	9.70e+1(1.87e+1)	6.85e+1(5.68e+0)–	8.62e+1(1.97e+1)–	6.89e+1(6.94e+0)–	1.10e+2(3.65e+1)≈
DTLZ2	1.43e-1(2.29e-2)	5.08e-1(7.66e-2)+	5.20e-1(8.87e-2)+	5.05e-1(8.64e-2)+	4.28e-1(6.68e-2)+
DTLZ3	1.97e+2(1.64e+1)	1.77e+2(9.53e+0)≈	2.34e+2(9.16e+1)≈	1.80e+2(7.04e+0)≈	2.72e+2(6.59e+1)+
DTLZ4	4.44e-1(1.34e-1)	4.89e-1(8.59e-2)+	5.30e-1(1.24e-1)+	5.01e-1(1.01e-1)+	6.45e-1(1.24e-1)+
DTLZ5	1.13e-1(2.24e-2)	4.70e-1(8.35e-2)+	4.26e-1(8.13e-2)+	4.86e-1(7.20e-2)+	3.02e-1(7.63e-2)+
DTLZ6	1.11e+0(5.70e-1)	4.03e-1(2.94e-1)–	6.83e-1(5.27e-1)–	2.75e-1(2.09e-1)–	5.71e+0(6.73e-1)+
DTLZ7	2.47e+0(1.89e+0)	3.83e-1(1.10e-1)–	7.56e-1(7.03e-1)–	5.27e-1(1.02e-1)–	7.12e+0(1.77e+0)+
+ / ≈ / –	- / - / -	3 / 1 / 3	3 / 1 / 3	3 / 1 / 3	6 / 1 / 0

F Experiments on Extremely Expensive Multi-Objective Optimization

In this section, we investigate the performance of our FSEO framework in the context of extremely expensive optimization, where the allowed fitness evaluations on target problems are fewer than that in the experiment carried out in Sections 5.1 and 5.2.1 of the main file.

F.1 Performance between Comparison Algorithms

We conduct the experiment described in Section 5.1 of the main file, but with a smaller evaluation budget than the budget listed in Table 6. The size of the initial dataset \mathcal{S}_* is set to 10, 60 for our MOEA/D-FS and comparison algorithms, respectively. 30 extra evaluations for further optimization are allowed. The total evaluation budget is 40, 90 for our MOEA/D-FS and comparison algorithms, respectively.

The aim of this subsection is to answer the question below:

- Is our FSEO framework more suitable for the optimization problems in which evaluations are extremely expensive? In other words, will the advantage of our FSEO framework become more prominent if the optimization problems allow a smaller evaluation budget?

The comparison results reported in Fig. 7 and Table 11 show that MOEA/D-FS has achieved competitive or smaller IGD+ values than MOEA/D-EGO on all DTLZ problems except for DTLZ7. Meanwhile, 5*d* evaluations have been saved.

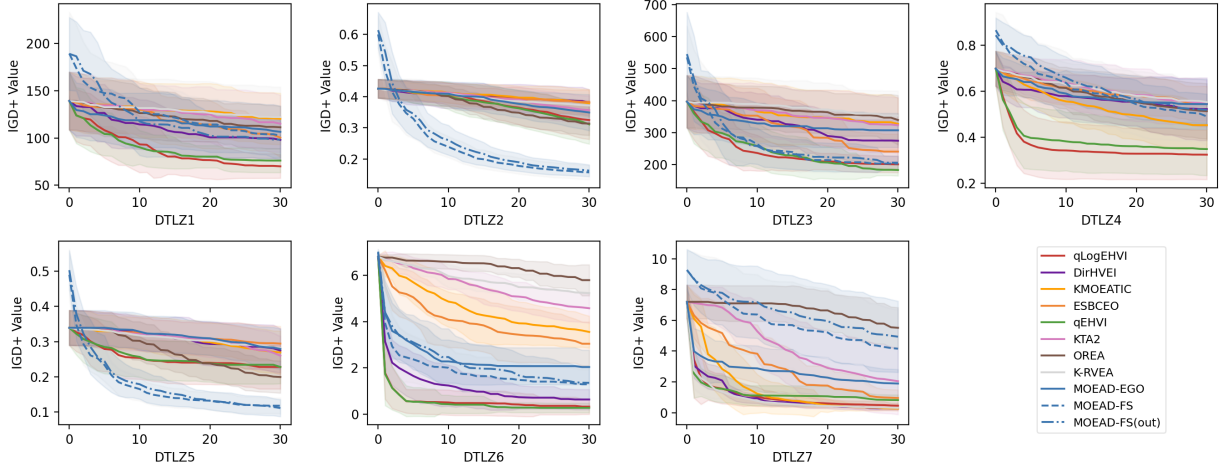


Figure 7: IGD+ curves averaged over 30 runs on 7 DTLZ problems. Solid lines are mean values, while shadows are error regions. MOEA/D-FSs and comparison algorithms initialize their surrogates with 10, 60 samples, respectively. Extra 30 evaluations are allowed in the further optimization. Note that ‘FS(out)’ indicates the target task is excluded from the range of related tasks during the meta-learning procedure. X-axis denotes the number of evaluations used after the surrogate initialization. In comparison to MOEA/D-EGO, both MOEA/D-FSs achieve smaller or competitive IGD+ values on all DTLZ test problems except for DTLZ7, while 50 evaluations are saved with the assistance from related tasks. Moreover, MOEA/D-FSs achieve the smallest IGD+ values on DTLZ2, DTLZ3, and DTLZ5.

Table 11: Mean IGD+ values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS and the comparison algorithms initialize their surrogates with 10, 60 samples, respectively. Extra 30 evaluations are allowed in the further optimization. Performance improvement can be observed from the comparisons between MOEA/D-FS and MOEA/D-EGO, while 50 evaluations are saved from surrogate initialization.

Problems	MOEA-D-EGO	MOEA-D-FS	qLogEHVI	DirHVEI	KMOEATIC	ESBCEO	qEHVI	KTA2	OREA	K-RVEA
DTLZ1	1.07e+2(2.73e+1)≈	1.03e+2(2.34e+1)	7.01e+1(1.25e+1)≈	9.80e+1(2.10e+1)≈	1.20e+2(2.71e+1)+	1.00e+2(2.07e+1)≈	7.62e+1(1.26e+1)≈	1.15e+2(3.03e+1)≈	1.11e+2(2.25e+1)+	1.22e+2(3.20e+1)+
DTLZ2	3.49e-1(5.82e-2)+	1.57e-1(2.29e-2)	3.25e-1(6.18e-2)+	3.81e-1(4.25e-2)+	3.79e-1(4.46e-2)+	3.83e-1(3.83e-2)+	3.11e-1(6.31e-2)+	3.57e-1(4.60e-2)+	3.14e-1(3.76e-2)+	3.72e-1(4.32e-2)+
DTLZ3	3.07e+2(5.32e+1)+	2.03e+2(2.42e+1)	2.01e+2(2.56e+1)≈	2.75e+2(6.57e+1)+	3.27e+2(8.10e+1)+	2.41e+2(5.51e+1)+	1.83e+2(1.82e+1)≈	3.23e+2(8.67e+1)+	3.39e+2(7.72e+1)+	3.53e+2(7.76e+1)+
DTLZ4	5.45e-1(1.09e-1)≈	4.91e-1(1.24e-1)	3.24e-1(1.08e-1)≈	5.23e-1(1.36e-1)≈	4.53e-1(1.03e-1)≈	5.47e-1(7.55e-2)≈	3.49e-1(1.15e-1)≈	5.47e-1(1.02e-1)≈	5.14e-1(1.21e-1)≈	5.53e-1(9.79e-2)≈
DTLZ5	2.79e-1(5.69e-2)+	1.18e-1(2.25e-2)	2.28e-1(6.23e-2)+	2.75e-1(6.28e-2)+	2.69e-1(6.11e-2)+	2.94e-1(4.92e-2)+	2.28e-1(4.70e-2)+	2.60e-1(5.50e-2)+	1.99e-1(4.53e-2)+	2.82e-1(5.42e-2)+
DTLZ6	2.04e+0(7.33e-1)+	1.29e+0(6.44e-1)	3.27e-1(3.32e-1)≈	6.35e-1(4.37e-1)≈	3.55e+0(6.90e-1)+	3.04e+0(9.46e-1)+	2.73e-1(2.08e-1)+	4.58e+0(6.36e-1)+	5.79e+0(6.70e-1)+	5.23e+0(6.17e-1)+
DTLZ7	1.90e+0(9.19e-1)≈	4.16e+0(2.54e+0)	4.57e-1(1.58e-1)≈	2.79e-1(1.17e-1)≈	2.68e-1(1.47e-1)≈	9.57e-1(5.40e-1)≈	8.23e-1(3.44e-1)≈	2.05e+0(2.16e+0)≈	5.51e+0(1.32e+0)+	3.13e-1(6.07e-2)≈
+ / ≈ / -	4/2/1	-/-	2/1/4	3/2/2	5/1/1	4/2/1	2/1/4	4/2/1	6/1/0	5/1/1

Consistent with the results discussed in Section 5.1 of the main file, MOEA/D-FS fails to achieve a competitive result compared to MOEA/D-EGO on DTLZ7 since experience is learned from small datasets collected from related tasks. Although we set a different evaluation budget for all SAEAs, the size of datasets for meta-learning $|\mathcal{D}_m|$ has not been modified. However, it can be observed from the statistical test results (see the last row of Tables 7 and 11) that our MOEA/D-FS outperforms the comparison algorithms on 30, 35 test instances when the total evaluation budget of comparison algorithms is set to 150, 90, respectively. This answers the question we raised before: The advantage of our FSEO framework is more prominent in the extremely expensive problems where a smaller evaluation budget is allowed. The comparison between the results obtained from Tables 7 and 11 has demonstrated that our FSEO framework is preferable when solving optimization problems within a very limited evaluation budget.

F.2 Out-Of-Range Analysis on Extremely Expensive Optimization

In Section 5.2.1 of the main file, we carried out an experiment to study the influence of task similarity on the performance of experience-based expensive multi-objective optimization. The optimization results obtained from the ‘in-range’ and the ‘out-of-range’ situations are compared. In this subsection, we conduct an experiment to investigate the difference between the ‘in-range’ and the ‘out-of-range’ situations for extremely expensive multi-objective optimization. The experimental setups are the

same as the setups described in Section 5.2.1 of the main file, except the allowed fitness evaluation budget is the same as described in Appendix F.1.

Table 12 gives the statistical test results, it can be seen that the ‘out-of-range’ situation achieves competitive IGD+ values to the ‘in-range’ situation on all 7 test instances. In comparison to MOEA/D-EGO, the experience gained in the ‘out-of-range’ situation leads to competitive or smaller IGD+ values on 6 DTLZ problems. Furthermore, similar results can be observed in the last row of Table 12, the ‘out-of-range’ situation achieves better/competitive/worse IGD+ values than all compared SAEAs on 34/14/15 test instances. In comparison, the ‘in-range’ situation achieves better/competitive/worse IGD+ values than all compared SAEAs on 35/13/15 test instances. There is only a minor difference between the optimization results obtained in two situations. These observations are consistent with the conclusions we made in Section 5.2.1 of the main file.

Table 12: Mean IGD+ values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. ‘Out-of-range’ indicates the target task is excluded from the range of related tasks during the meta-learning procedure. Both MOEA/D-FSs initialize their surrogates with 10 samples, extra 30 evaluations are allowed in the further optimization. ‘+’, ‘ \approx ’, and ‘-’ denote the result of the ‘out-of-range’ situation is statistically significantly superior to, almost equivalent to, and inferior to that of the ‘in-range’ situation in the Wilcoxon rank sum test (significance level is 0.05), respectively. The last two rows count the statistical test results between MOEA/D-FSs and other compared algorithms.

MOEA/D-FSs	In-range	Out-of-range
DTLZ1	1.03e+2(2.34e+1) \approx	9.84e+1(2.04e+1)
DTLZ2	1.57e-1(2.29e-2) \approx	1.62e-1(1.90e-2)
DTLZ3	2.03e+2(2.42e+1) \approx	2.06e+2(2.13e+1)
DTLZ4	4.91e-1(1.24e-1) \approx	5.20e-1(6.92e-2)
DTLZ5	1.18e-1(2.25e-2)+	1.11e-1(2.41e-2)
DTLZ6	1.29e+0(6.44e-1) \approx	1.36e+0(7.36e-1)
DTLZ7	4.16e+0(2.54e+0) \approx	4.94e+0(2.31e+0)
+ / \approx / -	0/7/0	-/-/-
vs MOEA/D-EGO	4/2/1	4/2/1
vs 9 Comparisons	35/13/15	34/14/15

F.3 Experimental Results in IGD and HV Metrics

Results in terms of IGD values are reported in Table 13 and Fig. 8. A smaller IGD value indicates a better optimization result.

Table 13: Mean IGD values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS and comparison algorithms initialize their surrogates with 10, 60 samples, respectively. Extra 30 evaluations are allowed in the further optimization.

Problems	MOEA-D-EGO	MOEA-D-FS	qLogEHVI	DirHVEI	KMOEAIC	ESBCEO	qEHVI	KPA2	OREA	K-RVEA
DTLZ1	1.07e+2(2.73e+1) \approx	1.03e+2(2.34e+1)	7.01e+1(1.23e+1)-	9.80e+1(2.10e+1) \approx	1.20e+2(2.71e+1)+	1.00e+2(2.07e+1) \approx	7.62e+1(1.26e+1)-	1.15e+2(3.03e+1) \approx	1.11e+2(2.25e+1)-	1.22e+2(3.20e+1)+
DTLZ2	3.69e-1(5.72e-2)+	1.91e-1(2.19e-2)	3.67e-1(4.92e-2)+	4.19e-1(2.95e-2)+	4.07e-1(3.85e-2)+	4.05e-1(3.07e-2)+	3.54e-1(5.44e-2)+	3.80e-1(4.24e-2)+	3.38e-1(3.44e-2)+	3.96e-1(3.55e-2)+
DTLZ3	3.07e+2(5.32e+1)+	2.03e+2(2.42e+1)	2.01e+2(2.56e+1) \approx	2.75e+2(6.57e+1)+	3.27e+2(8.10e+1)+	2.41e+2(5.51e+1)+	1.83e+2(1.82e+1) \approx	3.23e+2(8.67e+1)+	3.39e+2(7.72e+1)+	3.53e+2(7.76e+1)+
DTLZ4	8.36e-1(1.51e-1) \approx	8.47e-1(1.87e-1)	4.37e-1(1.04e-1)-	7.46e-1(1.35e-1)-	5.97e-1(1.23e-1)-	7.68e-1(1.21e-1)-	4.64e-1(1.04e-1)-	7.93e-1(1.49e-1) \approx	7.89e-1(1.67e-1) \approx	7.28e-1(1.16e-1)-
DTLZ5	2.88e-1(5.64e-2)+	1.22e-1(2.10e-2)	2.98e-1(4.43e-2)+	3.10e-1(4.79e-2)+	2.93e-1(5.32e-2)+	3.10e-1(4.29e-2)+	2.96e-1(3.01e-2)+	2.73e-1(5.06e-2)+	2.12e-1(4.27e-2)+	2.99e-1(5.02e-2)+
DTLZ6	2.08e+0(7.16e-1)+	1.36e+0(6.03e-1)	5.72e-1(3.24e-1)-	8.84e-1(3.53e-1)-	3.57e+0(6.85e-1)+	3.10e+0(8.82e-1)+	5.57e-1(2.05e-1)-	4.58e+0(6.36e-1)+	5.79e+0(6.67e-1)+	5.24e+0(6.15e-1)+
DTLZ7	2.02e+0(8.97e-1)-	4.22e+0(2.52e+0)	5.43e-1(1.62e-1)-	4.11e-1(1.26e-1)-	3.59e-1(1.49e-1)-	1.02e+0(5.29e-1)-	9.01e-1(3.21e-1)-	2.12e+0(2.13e+0)-	5.53e+0(1.32e+0)+	4.03e-1(7.19e-2)-
+ / \approx / -	4/2/1	-/-	2/1/4	3/1/3	5/0/2	4/1/2	2/1/4	4/2/1	6/1/0	5/0/2

Results in terms of HV values are reported in Table 14 and Fig. 9. A larger HV value indicates a better optimization result.

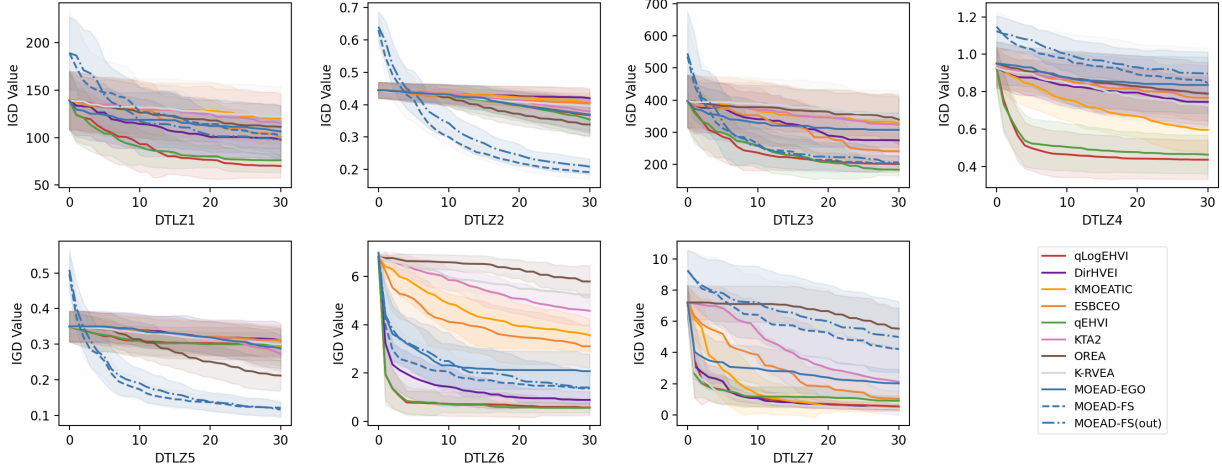


Figure 8: IGD curves averaged over 30 runs on 7 DTLZ problems. Solid lines are mean values, while shadows are error regions. MOEA/D-FSs and comparison algorithms initialize their surrogates with 10, 60 samples, respectively. Extra 30 evaluations are allowed in the further optimization. Note that ‘FS(out)’ indicates the target task is excluded from the range of related tasks during the meta-learning procedure. X-axis denotes the number of evaluations used after the surrogate initialization.

Table 14: Mean HV values and standard deviation (in parentheses) obtained from 30 runs on 7 DTLZ problems. MOEA/D-FS and comparison algorithms initialize their surrogates with 10, 60 samples, respectively. Extra 30 evaluations are allowed in the further optimization.

Problems	MOEA/D-EGO	MOEA/D-FS	qLogEHVI	DirHVEI	KMoeATIC	ESBCEO	qEHVI	KTA2	OREA	K-RVEA
DTLZ1	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈
DTLZ2	1.63e-1(8.93e-2)+	4.37e-1(3.48e-2)	2.30e-1(1.23e-1)+	7.68e-2(3.26e-2)+	9.43e-2(4.62e-2)+	1.25e-1(5.19e-2)+	2.62e-1(1.13e-1)+	1.25e-1(4.84e-2)+	1.73e-1(4.75e-2)+	1.05e-1(4.43e-2)+
DTLZ3	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈
DTLZ4	6.44e-2(6.93e-2)≈	1.00e-1(6.58e-2)	1.62e-1(1.24e-1)≈	5.40e-2(7.50e-2)+	4.77e-2(5.93e-2)+	1.55e-2(2.64e-2)+	1.37e-1(1.09e-1)+	2.18e-2(3.52e-2)+	5.58e-2(6.13e-2)+	2.28e-2(4.11e-2)+
DTLZ5	2.62e-2(2.46e-2)+	1.60e-1(1.54e-2)	1.63e-2(2.26e-2)+	1.01e-2(2.08e-2)+	2.04e-2(2.38e-2)+	1.43e-2(1.32e-2)+	1.83e-2(2.44e-2)+	2.60e-2(1.91e-2)+	4.57e-2(2.76e-2)+	1.51e-2(1.58e-2)+
DTLZ6	3.82e-4(2.06e-3)	1.07e-2(2.64e-2)	1.10e-1(3.55e-2)-	3.00e-2(4.66e-2)-	0.00e+0(0.00e+0)≈	8.07e-3(3.02e-2)≈	1.03e-1(4.71e-2)-	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈	0.00e+0(0.00e+0)≈
DTLZ7	6.98e-2(1.00e-1)≈	4.14e-2(8.25e-2)	1.97e-1(9.32e-2)-	4.15e-1(3.70e-2)-	3.38e-1(4.00e-2)-	8.06e-2(8.74e-2)-	1.04e-1(8.15e-2)-	1.31e-1(1.20e-1)-	0.00e+0(0.00e+0)≈	2.65e-1(3.94e-2)-
+/-	2/50	4/4	2/32	3/22	3/31	3/31	2/32	3/31	3/40	3/31

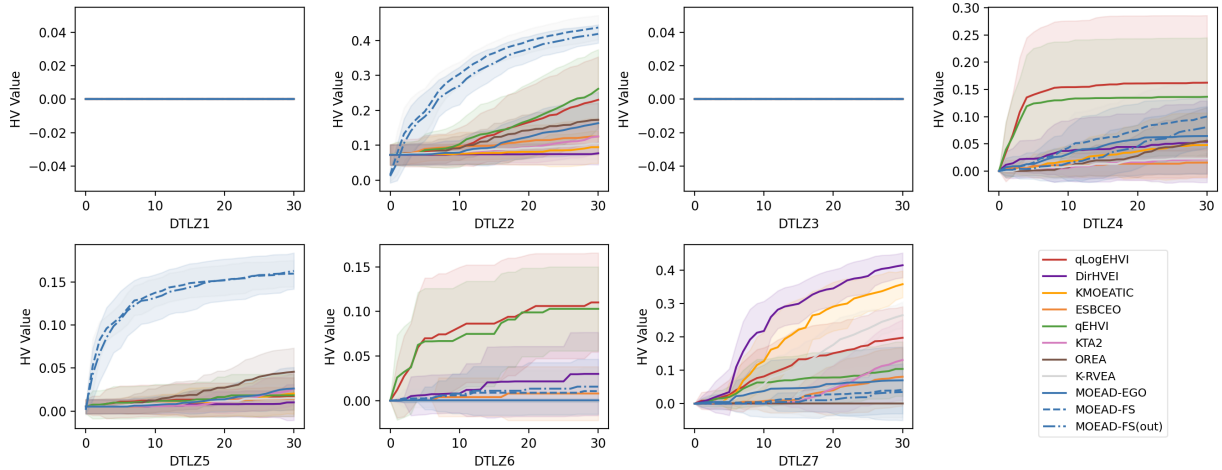


Figure 9: HV curves averaged over 30 runs on 7 DTLZ problems. Solid lines are mean values, while shadows are error regions. MOEA/D-FSs and comparison algorithms initialize their surrogates with 10, 60 samples, respectively. Extra 30 evaluations are allowed in the further optimization. Note that ‘FS(out)’ indicates the target task is excluded from the range of related tasks during the meta-learning procedure. X-axis denotes the number of evaluations used after the surrogate initialization.

G Performance on Real-World Network Architecture Search Problem

In this section, we demonstrate the performance of our FSEO framework on a real-world EMOP: network architecture search (NAS). NAS is a popular multi-objective optimization problem in the machine learning community. We use a NAS benchmark [25] and consider the architecture optimization of a Transformer network with two objectives: error and flops. The considered NAS problem has $d=18$ variables, each variable denotes the architecture of a network node. Related tasks are other Transformer architecture optimization problems with different architecture setups. The comparison algorithms are the same as described in Appendix E.2.

G.1 Experimental Setups

In the meta-learning procedure, both the support set and the query set contain 18 data points, thus $|\mathcal{D}_m|=36$. For MOEA/D-FS, 18 samples from the target task are used to initialize surrogates, while other comparison algorithms use 100 samples for initialization. For all algorithms, extra 50 evaluations are allowed as optimization budget (the x-axis in Fig. 4). Therefore, the total evaluation budgets for MOEA/D-FS and other comparison algorithms are 68 and 150, respectively. The remaining setups are the same as listed in Table 6.

H Performance on Real-World Engine Calibration Problem

To demonstrate algorithm generality and broad applicability, we also test FSEO on a real-world constrained optimization problem, showing that sampling efficiency can be enhanced in both the objective space and the constraint space.

The problem is a gasoline motor engine calibration problem with 6 adjustable engine parameters, namely throttle angle, waste gate orifice, ignition timing, valve timings, state of injection, and air-fuel-ratio. The calibration aims at minimizing BSFC while satisfying 4 constraints in terms of temperature, pressure, CA50, and load simultaneously.

H.1 Comparison Algorithms

Since the comparison algorithms in the DTLZ optimization experiments are not designed for handling constrained optimization, our comparison is conducted with 3 state-of-the-art constrained optimization algorithms used in industry [53]: A variant of EGO designed to handle constrained optimization problems (cons_EGO), a GA customized for this calibration problem (adaptiveGA), and a bilevel constrained SAEA (SAB-DE). The settings of the comparison algorithms are the same as suggested in the literature. In this experiment, we apply our FSEO framework to cons_EGO and investigate its optimization performance. Specifically, we meta-learn MDKL surrogates for each objective and each constraint separately, and set the underlying optimization method as well as the constraint handling technique in cons_EGO as an optimizer in Fig. 1. The resulting algorithm is denoted as cons_FS in our experiments.

H.2 Experimental Setups

The setup of related tasks (N, \mathcal{D}_i) is the same as described in Appendix D. In the meta-learning procedure, both the support set and the query set contain 6 data points, thus $|\mathcal{D}_m| = 12$. The total evaluation budget for all algorithms is set to 60. For adaptiveGA, all evaluations are used in the optimization process as it is not an SAEA. For cons_EGO and SAB-DE, 40 samples are used to initialize the surrogates and 20 extra evaluations are used in the optimization process. For cons_FS, only 6 samples are used to initialize MDKL surrogates, and the remaining evaluations are used for further optimization.

H.3 Optimization Results and Analysis

From the left side of Fig. 10, it can be observed that the minimal BSFC obtained by cons_FS decreases drastically in the first few evaluations, implying that the experience learned from related tasks is effective. In comparison, the minimal BSFC obtained by adaptiveGA and cons_EGO drops in a relatively slow rate, even though cons_EGO has used 34 more samples to initialize its surrogates. The star marker denotes the point at which cons_FS has evaluated 20 samples after surrogate initialization. It is worth noting that when 20 samples have been evaluated in the optimization, cons_FS achieves a smaller BSFC value than cons_EGO. After the star marker, the decrease of BSFC becomes slow as cons_FS has reached the optimal region. Therefore, further improvement in the normalized BSFC value is not significant and thus hard to observe.

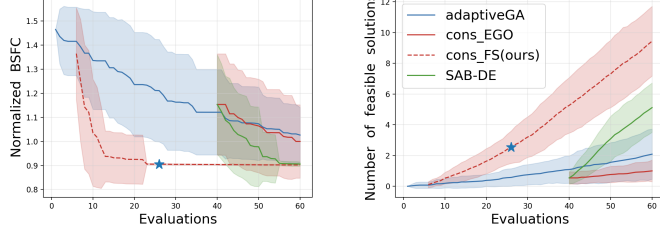


Figure 10: Engine Calibration results. Left figure shows optimization results. The star markers highlight the results achieved when 20 evaluations are used in the optimization process. Right figure illustrates the number of feasible solutions.

The advantages of our FSEO framework can also be observed in constraint handling. On the right side of Fig. 10, cons_FS finds more feasible solutions than the 3 comparison algorithms. These results indicate that our FSEO framework improves the performance of cons_EGO on both objective function and constraint functions. Meanwhile, only 1d evaluations are used to initialize surrogates.

Discussion on runtime. It should be noted that evaluating real engine performance on facilities is very costly in terms of both time and financial budget [55]. A single real engine performance evaluation can cost several hours [26], making the time cost of the meta-learning procedure negligible as it takes only a few minutes.

I Summary of Experiments

Our computational studies have demonstrated the following: First, we provide empirical evidence to show the effectiveness of learning experience: The meta-learning of neural network parameters and base kernel parameters are essential to the modeling accuracy of an MDKL model. As a result, our MDKL model outperforms the compared meta-learning modeling and non-meta-learning modeling methods on both the engine fuel consumption regression task and the sinusoid function regression task.

Second, we demonstrate the main contribution of this work via different EMOP benchmarks: In most situations, the proposed FSEO framework can assist regression-based SAEAs to reach competitive or even better optimization results, while the cost of surrogate initialization is only 1d samples. Due to the effectiveness of saving evaluations, our FSEO framework is preferable to other SAEAs when solving problems within a very limited evaluation budget. Moreover, some empirical guidelines are concluded to help the application of our FSEO framework. For the influence of task similarity, we find that related tasks that are very similar to the target task are not necessary to the application of our approach. The influence of these similar tasks on optimization performance is limited. Our FSEO framework can achieve competitive results without datasets from very similar related tasks. Besides, for the related tasks used for meta-learning, we have demonstrated that more useful experience can be learned if more data points are sampled from related tasks.

Third, the effectiveness of our FSEO framework is validated on two real-world optimization problems: A network architecture search problem and an engine calibration problem. Competitive or better results are achieved on the objective and constraint functions, while 1d samples are used to initialize surrogates. Therefore, our FSEO framework can also be applied to popular optimization scenarios such as constrained optimization, showing its generality and broad applicability.

J Time Complexity Analysis

The time complexity of our FSEO framework fluctuates with the underlying optimization algorithm it integrates with. However, the time complexity of the meta-learning procedure is fixed and we analyze it as follows:

Symbols (For simplicity, the symbols below are not consistent with those used in our paper.):

- N : the number of datasets for meta-learning.
- n : the size of each dataset for meta-learning.
- l : the number of neurons in each neural network layer.

Time complexities:

Our meta-learning model MDKL consists of GPs and Neural Networks. For GPs, the time complexities of each step are:

- Covariance matrix computation: $O(n^2)$
- Cholesky decomposition: $O(n^3)$
- Likelihood evaluation: $O(n^3)$
- Parameter update: $O(n^3 N)$

Taking neural networks into account:

- Network computation: $O(l^2)$

Therefore, the overall time complexity of the meta-learning procedure is: $O(n^3 N l^2)$

In the context of meta-learning, we have $N \gg n$ and l , hence, the time complexity of the meta-learning procedure mainly depends on the number of datasets for meta-learning N , which can be simplified as a linear complexity $O(N)$. As discussed in Section 4.4 and Appendix H, such a linear time complexity of meta-learning is negligible for expensive optimization problems.